RECEIVED
CENTRAL FAX CENTER
FEB 2 7 2007

AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application.

Claims 1-29 (Cancelled)

30. (Currently Amended) A machine-readable medium selected from the group consisting of a memory, a read only memory (ROM), a random access memory (RAM), a magnetic disk storage media, an optical storage media, and a flash memory device, that provides instructions, which when executed by a machine, cause the machine to perform operations comprising:

receiving a binary of a program code, the binary based on a first instruction set architecture;

checking one or more settable compatibility controls defined on top of a second instruction set architecture that have been set by a program environment, wherein the one or more settable compatibility controls indicate a compatibility with which the binary is to be translated; and

translating the binary to a translated binary based on the one or more settable compatibility controls, wherein the translated binary is based at least in part on [[a]] the second instruction set architecture, and wherein the translating is performed without fully preserving program semantics of the binary by deviating from the program semantics of the binary in exchange for improved execution performance of the translated binary according to the one or more settable compatibility controls.

31. (Cancelled)

32. (Previously Presented) The machine-readable medium of claim 30, wherein said translating comprises storing a portion of a hardware stack in a register of a processor translating the binary.

33. (Currently Amended) A method comprising:

receiving a binary of a program code, the binary based on a first instruction set architecture;

checking one or more settable compatibility modes ~~controls~~ that have been set by a programming environment to indicate a compatibility level with which to perform [[the]] a translation;

translating the binary to a translated binary that is based at least in part on a second instruction set architecture, wherein said translating includes using the compatibility level indicated by the one or more settable compatibility modes ~~controls~~, and wherein said translating includes deviating from precise program semantics of the binary in exchange for improved execution performance of the translated binary according to the one or more settable compatibility modes; and

executing the translated binary.

34. (Currently Amended) The method of claim 33, wherein the one or more settable compatibility modes ~~controls~~ control a level of semantic compatibility between the binary and the translated binary.

35. (Currently Amended) The method of claim 33, ~~wherein~~ further comprising an operating system setting the one or more compatibility modes ~~controls are set by an operating system~~.

36. (Currently Amended) The method of claim 33, ~~wherein~~ further comprising a user setting the one or more compatibility modes ~~controls are set by a user~~.

37. (Currently Amended) The method of claim 33, further comprising setting the one or more compatibility modes ~~controls~~ based on one or more command line flags associated with a command.

38. (Currently Amended) The method of claim 33, further comprising storing the one or more compatibility modes ~~controls~~ in a register of a processor translating the binary.

39. (Previously Presented) The method of claim 33, wherein the first instruction set architecture includes in-order accesses to memory and the second instruction set architecture includes out-of-order accesses to memory, the translating of the binary to include out-of-order accesses to memory by a processor executing the binary.

40. (Previously Presented) The method of claim 33, wherein the first instruction set architecture allows for self-modifying code and the second instruction set architecture does not allow for self-modifying code, the translating of the binary to include providing an instruction to controllers of memories that store the binary to perform write operations independent of checks of whether the write operations modify a location where the binary is stored.

41. (Previously Presented) The method of claim 33, wherein the second instruction set architecture has an address space that is larger than the first instruction set architecture, and wherein the translating of the binary comprises using the address space of the second instruction set architecture.

42. (Previously Presented) The method of claim 33, wherein data accessed by the binary is stored in a single segment in memory, and wherein an offset value for translating a virtual address to a physical address for the data is not modified during execution of the binary.

43. (Previously Presented) A system comprising:

a dynamic random access memory to include a binary of a program code based on a first instruction set architecture;

a processor coupled to the dynamic random access memory to translate the binary to a translated binary that is based at least in part on a second instruction set architecture, wherein the processor is to translate the binary by deviating from precise semantics of the binary in exchange for advantages offered by the second instruction set architecture based on one or more settable controls that have been set by a programming environment to control the deviation.

44. (Cancelled)

45. (Currently Amended) The system of claim 43, wherein the processor ~~comprises a register~~ is to store the settable controls in a register.

46. (Cancelled)

47. (Currently Amended) The system of claim 43, wherein the second instruction set architecture has an address space that is larger than the first instruction set architecture, and wherein the translated binary uses ~~translating of the binary comprises using~~ the address space of the second instruction set architecture.

48. (Previously Presented) The system of claim 43, wherein the binary is stored in a single segment in the memory, and wherein an offset value for translating a virtual address to a physical address is not modified during execution of the binary.

49. (Currently Amended) An apparatus comprising:

at least one register to store at least one settable flag, the at least one settable flag being settable by a program environment to control a compatibility level of a translation of a binary based on a first instruction set architecture to a translated binary based in part on a second instruction set architecture;

a decoder to receive the binary and to check the at least one settable flag in the at least one register, the decoder to translate the binary based at least in part on the settable flag, wherein the decoder is to deviate from precise program semantics of the binary in exchange for improved execution performance of the translated binary according to the at least one settable flag.

50. (Previously Presented) The apparatus of claim 49, wherein the at least one settable flag controls a level of semantic compatibility with which to translate the binary.

51. (Previously Presented) The apparatus of claim 49, wherein the at least one settable flag is settable by a user.

52. (Previously Presented) The apparatus of claim 49, wherein the translating of the binary comprises storing a portion of a hardware stack in a register of the at least one register.

53. (Previously Presented) The apparatus of claim 49, wherein the apparatus is coupled to a memory to store the binary, wherein the first instruction set architecture allows for self-modifying code and the second instruction set architecture does not allow for self-modifying code, the translating of the binary to include an instruction to a controller of the memory to cause the memory controller not to check whether code is self modifying.

54. (Previously Presented) The apparatus of claim 49, wherein the second instruction set architecture has an address space that is larger than the first instruction set architecture, and wherein the translating of the binary comprises using the address space of the second instruction set architecture.

55. (Previously Presented) The apparatus of claim 49, wherein data accessed by the binary is stored in a single segment in memory coupled to the apparatus, and wherein an offset value for translating a virtual address to a physical address for the data is not modified during execution of the binary.

56. (Currently Amended) A machine-readable medium selected from the group consisting of a memory, a read only memory (ROM), a random access memory (RAM), a

magnetic disk storage media, an optical storage media, and a flash memory device, that provides instructions, which when executed by a machine, cause the machine to perform operations comprising:

checking one or more settable ~~controls~~ compatibility modes that have been set by a programming environment to indicate a compatibility level with which to perform a translation of a binary based on a first instruction set architecture to a translated binary that is based at least in part on a second instruction set architecture; and

translating the binary to the translated binary, wherein said translating is based at least in part on the one or more settable ~~controls~~ compatibility modes, and wherein said translating is performed without fully preserving program semantics of the binary by deviating from the program semantics of the binary in exchange for improved execution performance of the translated binary according to the one or more settable compatibility modes.

57. (Currently Amended) The machine-readable medium of claim 56, wherein the one or more settable ~~controls~~ compatibility modes control a level of semantic compatibility between the binary and the translated binary.

58. (Currently Amended) The machine-readable medium of claim 56, wherein the ~~controls~~ compatibility modes are set by a user.

59. (Currently Amended) The machine-readable medium of claim 56, wherein said checking the ~~controls~~ compatibility modes comprises checking one or more command line flags associated with a command.

60. (Currently Amended) The machine-readable medium of claim 56, wherein said checking the ~~controls~~ compatibility modes comprises checking one or more registers of a processor.

61. (Previously Presented) The machine-readable medium of claim 56, wherein the first instruction set architecture allows for self-modifying code and the second instruction set architecture does not allow for self-modifying code, the translating of the binary to include an instruction to a controller of a memory to store the binary not to check whether a write operation modifies a location where the binary is stored.

62. (Currently Amended) A system comprising:

a dynamic random access memory to store a binary that is based on a first instruction set architecture that allows binaries to self modify;

a memory controller associated with the dynamic random access memory;

a translation logic coupled to the dynamic random access memory to receive the binary, the translation logic is to translate the binary to a translated binary that is based, at least in part, on a second instruction set architecture that does not allow binaries to self modify, wherein during the translation the translating logic is to instruct the memory controller to perform write operations without checking whether binary is self modifying.

63. (Previously Presented) The system of claim 62, wherein the translation logic is to translate the binary based on at least one settable control that is settable by a program environment.

64. (Currently Amended) The system of claim [[62]] 63, wherein the at least one settable control controls a level of semantic compatibility between the binary and the translated binary.

65. (Currently Amended) The system of claim [[62]] 63, wherein the at least one settable control allows the decoder to deviate from precise semantics of the binary in exchange for advantages offered by the second instruction set architecture.

66. (Cancelled)